C16/C116+4

COMPUTING-MONTHLY

ISSUE 8

NOVEMBER 1989

VOLUME 1

C16 / C116

SOON BE A5 FORMAT!

+4

UNDERGROUND PROG PART1

PETER CRACK'S TRAP THE KING PART 2

M/C SCROLL PROGRAM

LETTERS

ETC!!!

!!!!!!!!!! PLUS/4

Contact
Ray Robinson
102 Cliff Rd
Hornsea
N-Humberside
HU18 1SE
Tel (0964) 534611 after 6pm

Editorial

Well how about this then, The November issue at the end of November, I don't think thats bad actually, I know its lacking in content, but the Christmas issue should be full of goodies for everyone.

Simon Pollard of Goole, North Humberside, has asked me to, mention that he is in the process of doing an arcade game in BASIC, nice one Si, it is based on DOUBLE DRAGON II, the arcade game, and its called STREET PATROL, Simon assures me that he will be putting the C16/+4's BASIC capabilities to the test, hopefully, next month I will have a preview copy and more details about this game.

The reason why this issue is only 18 pages is becuase I did'nt recieve much articles, but Peter Crack managed to help out on that, with his 2nd part of Trap The King and his Scroll Routine, which Simon Pollard wanted help on, well Si, I hope Peter has filled your requirements (is that spelt right??), thanks to Peter again, from myself and Simon.

Guess what folks, no I have'nt won the pools, no I have'nt got my college grant, no I can't spell, but I've Blown Up My C+4, whata plonker, oh well its a good job I've 2 standby C16's, to fall back on, so if anybody is interested in the +4 then offer me a price, John Hadlow, I think your were interested, if yes write.

Well I've run out of gas, and can't think of anything else, except sos for the slim mag, but December will be better, and hopefully I have another Printer Ribbon. One last thing, go on have a guess at how many members the club now has, yep a staggering 35 and its growing, many thanks to the newest members for joining, but you have to agree, who else is supporting the C16/+4 like I'm trying to do, and don't forget its just me, and what will COBOL programming at college it can get a bit confusing and time consuming, so if your mag is ever late, please spare a thought, I'm slaving in my spare time to put together a mag, well the only mag for the C16/+4, so you must understand my position. I think I'll carry on talking, well members, I'm trying to set up a BOOK hire scheme, so if you have any C16/+4 computer books that you would sell (PLEASE NOT TO EXPENSIVE) then contact me please. I'm also trying to set up a PD software scheme of my own, but more details later.

Right I want your articles, because if you don't send, I can't print, and the mag may cease to exist, so please keep'um coming, I want any C16/+4 related articles, so send, send, send, PPPPPPLLLLLEEEEAAASSE!!!!!

Well folks what da ya think to the new front cover, very good eh? Well I thought so because I liked its original idea, but I need a few lessons on how to print out lettreset, but I'll get there, eventually. Many thanks to James McBride of Middlessex, I hope you like the tapes, any problems contact me.
One last thing, some of the members have been asking about the questionaires, re, REQUEST for them to be retunrned, New members need not worry about these, they were for those who joined in April 1989.

Bye for now Editor

****

.....TRAP THE KING PART TWO.....

****

LAST MONTHS SECTION COULD BE RUN AND VIEWED,NOT SO THIS MONTHS ADDITION, ONCE YOU HAVE TYPED IT IN SAVE IT!! IT WILL NOT RUN BECAUSE MOST OF THE SUB ROUTINES ARE MISSING (THEY WILL BE IN PART THREE).

THIS IS HOW THIS PART RUNS.

135E-1360 STORES THE NUMBER OF TYPES OF PIECES IN $D6=(#$01-#$04).

1362-1364 STORES THE INDIVIDUAL PIECE NUMBER IN $E7=(#$00-#$23).

1366-1368 STORES THE ASKEY CHARACTER NUMBER FOR THAT GROUP OF PIECES IN $D0 (#$68-#$6F) #$6B=YOUR INFANTRY.

136A-1372 SET DECIMAL MODE,THE COMPUTER NOW COUNTS IN BASE TEN (0-9).AND INCREASES THE TURN COUNTER BY ONE. THE POSSIBLE RANGE OF TURN NUMBERS IS 1-99.THEN STARTS AT ONE AGAIN.

1373-1382 PUSH COPY OF TURN COUNTER ON TO STACK,MOVE THE FOUR RIGHTMOST BITS IN THE BYTE TO THE LEFT FOUR PLACES CLEARING THE RIGHT FOUR BITS AS IT DOES SO,MIX IN #$30 TURNING IT INTO AN ASKEY CODE FOR A NUMBER,AND PRINT IT IN THE TENS COLUMN OF THE TURN COUNTER DISPLAY ON THE SCREEN.PULL THE TURN COUNTER BYTE OFF THE STACK,CLEAR THE FOUR LEFTMOST BITS RETAINING THE OTHERS,MIX IN #$30 AND STORE IT IN THE UNITS COLUMN,GIVING A DISPLAY FROM 01 TO 99.

1385-1392 GROUP OF NESTED FOR NEXT LOOP COUNTERS. LOAD X REGISTER WITH NUMBER OF DIFFERENT GROUPS OF PIECES,PUSH COPY ON THE STACK,LOAD Y REGISTER WITH NUMBER OF PIECES WITHIN GROUP,PUSH A COPY ON THE STACK,LOAD Y REGISTER WITH THE NUMBER OF MOVES FOR EACH PIECE.AND PUSH THIS ONTO THE STACK.

1393 GET POSSITION $D1 AND $D2 ARE THE REGISTERS THAT WILL CONTAIN THE COORDS $D2 CONTAINS THE UP AND DOWN COORDINATE

```
. 135E  A9 04     LDA #$04
. 1360  85 D6     STA $D6
. 1362  A2 00     LDX #$00
. 1364  86 E7     STX $E7
. 1366  A9 6B     LDA #$6B
. 1368  85 D0     STA $D0
. 136A  F8        SED
. 136B  18        CLC
. 136C  A5 D9     LDA $D9
. 136E  69 01     ADC #$01
. 1370  85 D9     STA $D9
. 1372  D8        CLD
. 1373  48        PHA
. 1374  4A        LSR
. 1375  4A        LSR
. 1376  4A        LSR
. 1377  4A        LSR
. 1378  09 30     ORA #$30
. 137A  8D C5 0F  STA $0FC5
. 137D  68        PLA
. 137E  29 0F     AND #$0F
. 1380  09 30     ORA #$30
. 1382  8D C6 0F  STA $0FC6
. 1385  A6 D6     LDX $D6
. 1387  8A        TXA
. 1388  48        PHA
. 1389  BC 57 22  LDY $2257,X
. 138C  98        TYA
. 138D  48        PHA
. 138E  BC 5B 22  LDY $225B,X
. 1391  98        TYA
. 1392  48        PHA
. 1393  20 A2 20  JSR $20A2
. 1396  A5 D2     LDA $D2
. 1398  D0 04     BNE $139E
. 139A  68        PLA
. 139B  4C 43 14  JMP $1443
. 139E  20 B2 20  JSR $20B2
. 13A1  20 00 1E  JSR $1E00
. 13A4  20 00 1F  JSR $1F00
```

IF A PIECE IS MISSING THEN BOTH WILL CONTAIN ZERO

1396-139B CHECK IF PEICE IS MISSING IF YES THEN GOTO COMPUTERS MOVE

139E MOVE CURSOR TO NEW POSSITION

13A1 CHECK ADJACENT SQUARES FOR ENEMY DIAGONALLY AND REVEAL CONTENTS.

13A4 DO THE SAME FOR ADJACENT SQUARES UP,DOWN,LEFT AND RIGHT (SQUARE CHECK).

13A7 THERE ARE LESS THAN TWO COMPUTER PIECES NEXT TO THIS ONE.

13A9 THERE IS MORE THAN ONE COMPUTER PIECE NEXT TO THIS ONE.

13AC RESET CURSOR TO YOUR PIECES COORDS

13AF-13BE STORE YOUR COLOUR IN COLOUR REGISTER,PRINT COLOUR FLAHS IN THIS SQUARE,REPRINT YOUR CHARACTER IN THIS SQUARE TO MAKE IT FLASH. RETURN CURSOR TO YOUR SQUARE (EVERY TIME JSR$FFD2 IS CALLED AND A PRINTABLE ASKEY CODE IS USED THE CURSOR IS AUTOMATICALLY MOVED ON ONE CHARACTER SQUARE,HENCE THIS ROUTINE).

13C1 CHECK IF MOVING PIECE IS A GUN.

13C4-13CA STORE A COPY OF PRESENT (OLD) COORDINATES IN $D3,$D5.

13CC-13D2 LOAD OLD COORDINATES FROM $D3 $D5 TO $D1,$D2 THIS MAY SOUND SILLY BUT IF THE ATTEMPTED NEXT MOVE CANNOT BE MADE THE PROGRAMME RETURNS TO $13CC AND LETS YOU TRY AGAIN.

13D4-13E0 CHECK IF 'ESC' KEY HAS BEEN PRESSED,IF SO END PROGRAMME RIGHT NOW. DO NOT USE THIS KEY UNLESS YOU ARE FED UP WITH THE GAME.

13E6-140C CHECK FOR THE 'E' 'D' 'S' 'X' AND ZERO KEYS,AND UPDATE $D1,$D2 COORDINATES ACCORDINGLY.

140E-1410 IF THE ZERO KEY WAS USED PULL THE NUMBER OF MOVES COUNTER OFF THE STACK,DISCARD IT AND FORCE BRANCH TO COMPUTERS MOVE.

1412 MOVE CURSOR TO NEW SQUARE.

1415 CHECK TO SEE IF IT IS EMPTY.

1418 NO ITS NOT SO TRY AGAIN.

```
. 13A7  90 03    BCC $13AC
. 13A9  4C 70 1E JMP $1E70
. 13AC  20 B2 20 JSR $20B2
. 13AF  A9 53    LDA #$53
. 13B1  8D 3B 05 STA $053B
. 13B4  A9 82    LDA #$82
. 13B6  20 D2 FF JSR $FFD2
. 13B9  A5 D0    LDA $D0
. 13BB  20 D2 FF JSR $FFD2
. 13BE  20 B2 20 JSR $20B2
. 13C1  20 80 1F JSR $1F80
. 13C4  A5 D1    LDA $D1
. 13C6  85 D3    STA $D3
. 13C8  A5 D2    LDA $D2
. 13CA  85 D5    STA $D5
. 13CC  A5 D3    LDA $D3
. 13CE  85 D1    STA $D1
. 13D0  A5 D5    LDA $D5
. 13D2  85 D2    STA $D2
. 13D4  20 E4 FF JSR $FFE4
. 13D7  F0 FB    BEQ $13D4
. 13D9  C9 1B    CMP #$1B
. 13DB  D0 09    BNE $13E6
. 13DD  20 00 20 JSR $2000
. 13E0  00       BRK
. 13E1  EA       NOP
. 13E2  EA       NOP
. 13E3  EA       NOP
. 13E4  EA       NOP
. 13E5  EA       NOP
. 13E6  C9 45    CMP #$45
. 13E8  D0 05    BNE $13EF
. 13EA  C6 D2    DEC $D2
. 13EC  38       SEC
. 13ED  B0 23    BCS $1412
. 13EF  C9 44    CMP #$44
. 13F1  D0 05    BNE $13F8
. 13F3  E6 D1    INC $D1
. 13F5  38       SEC
. 13F6  B0 1A    BCS $1412
. 13F8  C9 53    CMP #$53
. 13FA  D0 05    BNE $1401
. 13FC  C6 D1    DEC $D1
. 13FE  38       SEC
. 13FF  B0 11    BCS $1412
```

141A YES IT IS SO MOVE THERE.

141D-142B MOVE CURSOR TO OLD POSSITION (SQUARE JUST LEFT) PRINT UNFLASH AND SPACE CHARACTERS.

142E-1433 PULL MOVE COUNTER OFF STACK, DECREASE IT,IF IT IS NOW ZERO THEN THAT WAS THE LAST MOVE,IF NOT THEN BRANCH BACK AND MOVE AGAIN.

HINT....IF YOU WANT TO MOVE YOUR PIECE NEXT TO TWO OR MORE COMPUTERS PIECES DO IT ON ITS LAST MOVE AS THE CHECK AROUND ROUTINE IS ONLY DONE AT THE BEGINING OF EACH MOVE!!!!

1436-1440 MOVE CURSOR TO NEXT POSSITION PRINT UNFLASH AND YOUR CHARACTERS.

1443-144F CHANGE POINTERS TO COMPUTERS VALUES (START OF COMPUTERS MOVE).

1451-1457 LOAD NUMBER OF MOVES FOR THIS PIECE AND PUSH A COPY ONTO THE STACK

1458 GET POSSITION.

145B-145D IS IT MISSING?

145F YES IT IS.

1462 NO IT IS NOT!,SO MOVE CURSOR TO THE CORRECT POSSITION.

1465-146F PRINT SPACE IN THAT POSSITION AND LOAD SCREEN COLOUR INTO COLOUR REG. REMEMBER,WHEN THE COMPUTER IS MOVING A PIECE WETHER IT CHANGES POSSITION OR NOT IT ALWAYS DISSAPEARS.

1470 MOVE CURSOR BACK TO MOVING PIECES POSSITION.

1473-1484 CHANGE CONTENTS OF $20E8,$20E9 $20EA TO READ JMP$1DD0. $E0=NUMBER OF EMPTY SQUARE ADJACENT TO COMPUTERS MOVING PIECE (SQUARE ON NOT DIAGONAL)

1486 CHECK AROUND (SQUARE ON ONLY).

1489-1493 CHANGE CONTENTS OF $20E8, $20E9,$20EA TO THEIR ORIGINAL VALUES THIS ALLOWS THE ROUTINE BEGINING AT $1F00 TO BE USED BY BOTH YOURS AND THE CMPUTERS TURN ROUTINES.

1496-149C CHECK FOR NUMBER OF YOUR

```
. 1401  C9 58     CMP #$59
. 1403  D0 05     BNE $140A
. 1405  E6 D2     INC $D2
. 1407  38        SEC
. 1408  B0 08     BCS $1412
. 140A  C9 30     CMP #$30
. 140C  D0 BE     BNE $13CC
. 140E  68        PLA
. 140F  38        SEC
. 1410  B0 24     BCS $1436
. 1412  20 B2 20  JSR $20B2
. 1415  20 BC 20  JSR $20BC
. 1418  B0 B2     BCS $13CC
. 141A  20 78 20  JSR $2078
. 141D  A6 D5     LDX $D5
. 141F  A4 D3     LDY $D3
. 1421  20 B6 20  JSR $20B6
. 1424  A9 84     LDA #$84
. 1426  20 D2 FF  JSR $FFD2
. 1429  A9 20     LDA #$20
. 142B  20 D2 FF  JSR $FFD2
. 142E  68        PLA
. 142F  A8        TAY
. 1430  88        DEY
. 1431  F0 03     BEQ $1436
. 1433  4C 91 13  JMP $1391
. 1436  20 B2 20  JSR $20B2
. 1439  A9 84     LDA #$84
. 143B  20 D2 FF  JSR $FFD2
. 143E  A5 D0     LDA $D0
. 1440  20 D2 FF  JSR $FFD2
. 1443  A5 D0     LDA $D0
. 1445  38        SEC
. 1446  E9 04     SBC #$04
. 1448  85 D0     STA $D0
. 144A  A5 E7     LDA $E7
. 144C  18        CLC
. 144D  69 12     ADC #$12
. 144F  85 E7     STA $E7
. 1451  A6 D6     LDX $D6
. 1453  BC 5B 22  LDY $225B,X
. 1456  98        TYA
. 1457  48        PHA
. 1458  20 A2 20  JSR $20A2
. 145B  A5 D2     LDA $D2
```

PIECES NEXT TO COMPUTERS MOVING PIECE ($DF CONTAINS NUMBER OF).

149E-14A0 CHECK FOR ESCAPE ROUTES ($E0 CONTAINS NUMBER OF).

14A2-14A4 IS COMPUTERS KING TRAPPED?.

14A6 NO.

14A8-14AD YES!! MODIFY PRINT ROUTINE AT $1D00 PRINT KING AND ALL OTHER ACTIVE COMPUTER PIECES AND SURRENDER (YOU WIN).

14B0-14B2 PIECE TRAPPED IS NOT KING. PUT RTS IN $1EA5.

14B5 PIECE SURRENDERS!!

14B8-14BA PUT ORIGINAL VALUE BACK IN TO $1EA5.

14BD UPDATE SCREEN('ENEMY LEFT')DISPLAY.

14C0-14C6 COMPUTERS PIECE HAS'FOUND'ONE OF YOURS,SO,PUT 'FOUND' COORDINATES INTO $D7,$D8 SO THAT OTHER COMPUTER PIECES CA CAN COME AND HELP

14C8 IS MOVING PIECE COMPUTERS KING?. IF 'YES' THEN IT MUST CONTINUE TO MOVE.

14CB-14D1 SAVE PRESENT POSSITION FOR BACKTRACK AND ABORTED MOVE RELOADS.

14D8-1508 CHECK WHICH WAY COMPUTERS PIECE HAS TO MOVE TO BRING IT ALONGSIDE 'FOUND' PIECE AND TRY TO MOVE IN THAT DIRECTION,PROVIDED THIS DOES NOT MEAN MOVING BACK ON ITSELF (BACKTRACKING).

1510 IF ABOVE MOVE IS NOT POSSIBLE THEN GET 'PRESENT' COORDINATES.

1513-151D SET RND GENERATOR LIMITS AND GET RND NUMBER.

1520-152E MULTIPLY RND NUMBER BY TWO (ASL) MIX IN #$E0 AND STERE IT $1527 TO CREATE A INDIRECT JMP ADDRESS.THIS COMMAND WILL EXPECT TO FIND (IN THIS CASE) THE LOW HALF IN BYTE NUMBER $22E0 AND THE HIGH HALF OF THE ADDRESS IT IS GOING TO JUMP TO IN $22E1. IF YOU CHECK MEMORY LOCATIONS $22E0-$22E7 THEN YOU WILL FIND THE FOLLOWING HEX VALUES:- 29.15.31.15.39.15.41.15 THIS EQUALLS ADDRESSES $1529,$1531,$1539, AND $1541 THESE ARE THE START ADDRESSES OF THE FOUR POSSIBLE MOVE ROUTINES FOR THE INDIRECT JUMP COMMAND.

145D D0 03 BNE $1462

```
. 145F  4C 88 15  JMP $1588
. 1462  20 B2 20  JSR $20B2
. 1465  A9 20     LDA #$20
. 1467  20 D2 FF  JSR $FFD2
. 146A  A9 05     LDA #$05
. 146C  8D 3B 05  STA $053B
. 146F  EA        NOP
. 1470  20 B2 20  JSR $20B2
. 1473  A9 4C     LDA #$4C
. 1475  8D E8 20  STA $20E8
. 1478  A9 D0     LDA #$D0
. 147A  8D E9 20  STA $20E9
. 147D  A9 1D     LDA #$1D
. 147F  8D EA 20  STA $20EA
. 1482  A9 00     LDA #$00
. 1484  85 E0     STA $E0
. 1486  20 00 1F  JSR $1F00
. 1489  A9 20     LDA #$20
. 148B  8D E8 20  STA $20E8
. 148E  8D EA 20  STA $20EA
. 1491  A9 B2     LDA #$B2
. 1493  8D E9 20  STA $20E9
. 1496  A5 DF     LDA $DF
. 1498  F0 31     BEQ $14CB
. 149A  C9 02     CMP #$02
. 149C  90 22     BCC $14C0
. 149E  A5 E0     LDA $E0
. 14A0  D0 29     BNE $14CB
. 14A2  A5 E7     LDA $E7
. 14A4  C9 23     CMP #$23
. 14A6  D0 09     BNE $14B0
. 14A8  A9 C0     LDA #$C0
. 14AA  8D 04 1D  STA $1D04
. 14AD  4C 71 1D  JMP $1D71
. 14B0  A9 60     LDA #$60
. 14B2  8D A5 1E  STA $1EA5
. 14B5  20 71 1E  JSR $1E71
. 14B8  A9 E0     LDA #$E0
. 14BA  8D A5 1E  STA $1EA5
. 14BD  4C E9 1D  JMP $1DE9
. 14C0  A5 D1     LDA $D1
. 14C2  85 D7     STA $D7
. 14C4  A5 D2     LDA $D2
. 14C6  85 D8     STA $D8
. 14C8  20 C7 1D  JSR $1DC7
```

1529-1543 FOUR MOVE ROUTINES ONE WILL BE CHOSEN AT RANDOM (SEE ABOVE), BACKTRACKING IS CHECKED.

1546 IF THE RND MOVE CANNOT BE MADE GET ORIGINAL COORDINATES AND TRY.....

1549-1563 TO MAKE FIRST AVAILABLE MOVE STARTING WITH UP. THEN DOWN THEN LEFT THEN RIGHT BUT ALWAYS CHECK BACKTRACKING

1569-1581 ....STILL NO GOOD? O.K. TRY THE SAME AGAIN BUT DISREGARD BACKTRACK

1583 'CANNOT MOVE EH?'....WELL THEN... STAY PUT!!!.

1589-158D WAS IT LAST MOVE?.

1590-159E YES!! SO RESET POINTERS FOR PLAYERS MOVE,WHILST AT THE SAME TIME INCREASING $E7(PIECES INDIVIDUAL NUMBER) AT $144D WE ADDED HEX $$12 SO NOW WE SUBTRACT HEX $$11 NET GAIN=HEX $$01

15A2-15A9 WAS IT LAST PIECE IN THIS GROUP.

15AC-15B6 WAS IT LAST GROUP.

15B8 YES IT WAS SO START ANOTHER TURN.

15C0-15E6 ROUTINE TO PRINT (REVEAL) ALL REMAINING COMPUTER PIECES.

15E8-15EE ROUTINE FOR ENDING COMPUTER PIECES MOVE. JSR$1D8E GOES TO A ROUTINE WHICH,AS ITS LAST ACTIONS PULLS THE LAST TWO BYTES OFF THE STACK(THESE ARE THE LAST JSR COMMANDS RETURN ADDRESSES) AND THEN JUMPS TO $1588.

I HAVE USED THIS METHOD BECAUSE THE $1D8E ROUTINE IS ENTERED SEVERAL TIMES AND WAYS AND IT IS THE ONLY WAY I COULD KEEP THE STACK POINTER IN THE RIGHT PLACE,BEARING IN MIND THAT THE WHOLE ROUTINE FROM $135E TO $15BB IS A SERIES OF NESTED LOOPS.

O.K. THATS PART TWO COMPLETED AS ALWAYS EITHER PHONE ME, OR WRITE IN TO THE MAGAZINE.

I SHALL SUBMIT PART THREE FOR THE NOVEMBER ISSUE AND FOR DECEMBER PART ONE OF 'BLOOPING BUG' A SPRITE PROGRAMME FOR THE +4 ,C16 C116 WITH 64K.

END... PETER SPLACK

```
. 14CB  A5 D1     LDA $D1
. 14CD  85 E5     STA $E5
. 14CF  A5 D2     LDA $D2
. 14D1  85 E6     STA $E6
. 14D3  EA        NOP
. 14D4  EA        NOP
. 14D5  EA        NOP
. 14D6  EA        NOP
. 14D7  EA        NOP
. 14D8  A5 D7     LDA $D7
. 14DA  C5 D1     CMP $D1
. 14DC  F0 12     BEQ $14F0
. 14DE  10 08     BPL $14E8
. 14E0  C6 D1     DEC $D1
. 14E2  20 80 1D  JSR $1D80
. 14E5  38        SEC
. 14E6  B0 08     BCS $14F0
. 14E8  20 CB 1E  JSR $1ECB
. 14EB  E6 D1     INC $D1
```

```
. 14ED  20 80 1D  JSR $1D80
. 14F0  20 CB 1E  JSR $1ECB
. 14F3  A5 D8     LDA $D8
. 14F5  C5 D2     CMP $D2
. 14F7  F0 17     BEQ $1510
. 14F9  10 08     BPL $1503
. 14FB  C6 D2     DEC $D2
. 14FD  20 80 1D  JSR $1D80
. 1500  38        SEC
. 1501  B0 0D     BCS $1510
. 1503  20 CB 1E  JSR $1ECB
. 1506  E6 D2     INC $D2
. 1508  20 80 1D  JSR $1D80
. 150B  EA        NOP
. 150C  EA        NOP
. 150D  EA        NOP
. 150E  EA        NOP
. 150F  EA        NOP
. 1510  20 CB 1E  JSR $1ECB
. 1513  A9 00     LDA $$00
. 1515  8D 3D 20  STA $203D
. 1518  A9 04     LDA $$04
. 151A  8D 39 20  STA $2039
. 151D  20 30 20  JSR $2030
. 1520  0A        ASL
```

```
. 1521  Ø9 EØ     ORA #$EØ
. 1523  8D 27 15  STA $1527
. 1526  6C E2 22  JMP ($22E2)
. 1529  C6 D2     DEC $D2
. 152B  2Ø 8Ø 1D  JSR $1D8Ø
. 152E  38        SEC
. 152F  BØ 15     BCS $1546
. 1531  C6 D1     DEC $D1
. 1533  2Ø 8Ø 1D  JSR $1D8Ø
. 1536  38        SEC
. 1537  BØ ØD     BCS $1546
. 1539  E6 D1     INC $D1
. 153B  2Ø 8Ø 1D  JSR $1D9Ø
. 153E  38        SEC
. 153F  BØ Ø5     BCS $1546
. 1541  E6 D2     INC $D2
  1543  2Ø 8Ø 1D  JSR $1D8Ø
  1546  2Ø CB 1E  JSR $1ECB
  1549  C6 D2     DEC $D2
  154B  2Ø 8Ø 1D  JSR $1D8Ø
  154E  2Ø CB 1E  JSR $1ECB
  1551  C6 D1     DEC $D1
  1553  2Ø 8Ø 1D  JSR $1D8Ø
  1556  2Ø CB 1E  JSR $1ECB
  1559  E6 D1     INC $D1
  155B  2Ø 8Ø 1D  JSR $1D8Ø
  155E  2Ø CB 1E  JSR $1ECB
  1561  E6 D2     INC $D2
  1563  2Ø 8Ø 1D  JSR $1D8Ø
  1566  2Ø CB 1E  JSR $1ECB
  1569  C6 D2     DEC $D2
  156B  2Ø 68 1D  JSR $1D68
  156E  2Ø CB 1E  JSR $1ECB
  1571  C6 D1     DEC $D1
  1573  2Ø 68 1D  JSR $1D68
  1576  2Ø CB 1E  JSR $1ECB
  1579  E6 D1     INC $D1
  157B  2Ø 68 1D  JSR $1D68
  157E  2Ø CB 1E  JSR $1ECB
  1581  E6 D2     INC $D2
  1583  4C E8 15  JMP $15E8

. 1586  EA        NOP
. 1587  EA        NOP
. 1588  68        PLA
. 1589  A8        TAY
. 158A  88        DEY
. 158B  FØ Ø3     BEQ $159Ø
. 158D  4C 56 14  JMP $1456
. 159Ø  A5 DØ     LDA $DØ
. 1592  18        CLC
. 1593  69 Ø4     ADC #$Ø4
. 1595  85 DØ     STA $DØ
. 1597  A5 E7     LDA $E7
. 1599  38        SEC
. 159A  E9 11     SBC #$11
. 159C  85 E7     STA $E7
. 159E  EA        NOP
. 159F  EA        NOP
. 15AØ  EA        NOP
. 15A1  EA        NOP
. 15A2  68        PLA
. 15A3  A8        TAY
. 15A4  88        DEY
. 15A5  FØ Ø5     BEQ $15AC
. 15A7  A6 D6     LDX $D6
. 15A9  4C 8C 13  JMP $138C
. 15AC  68        PLA
. 15AD  AA        TAX
. 15AE  CA        DEX
. 15AF  FØ Ø7     BEQ $15B8
. 15B1  C6 D6     DEC $D6
. 15B3  C6 DØ     DEC $DØ
. 15B5  4C 87 13  JMP $1387
. 15B8  4C 5E 13  JMP $135E
. 15BB  EA        NOP
```

```
170 JJ$="[1 CD][5 CR][ORANGE][REV,ON]JUB
ILEE LINE[REV,OFF][WHITE]"

180 MM$="[1 CD][5 CR][PURPLE][REV,ON]MET
ROPOLITAN LINE[REV,OFF][WHITE]"

190 ES$="[1 CD][5 CR][PURPLE]METROPOLITA
N EAST LONDON SECTION[WHITE]"

200 NN$="[1 CD][5 CR][REV,ON]NORTHERN LI
NE[REV,OFF]"

210 PP$="[1 CD][5 CR][DARK BLUE][REV,ON]
PICCADILLY LINE[REV,OFF][WHITE]"

220 VV$="[1 CD][5 CR][LIGHT BLUE][REV,ON
]VICTORIA LINE[REV,OFF][WHITE]"

230 CH$="[1 CD][2 CR][RED]YOU CAN INTERC
HANGE WITH BRITISH RAIL[WHITE]"

240 LO$="[1 CD][17 CR][RED][SHIFT,M][1 C
D][2 CL][CBM,T 3 TIMES][1 CD][3 CL][CBM,
@ 3 TIMES][1 CD][2 CL][SHIFT,M][WHITE]"

250 BR$="[1 CD][5 CR][RED]BRITISH RAIL L
INK LINE[WHITE]"

260 PRINT"[CLEAR][12 CD][8 CR]PLEASE WAI
T READING DATA"

270 FORX=0TO270:READS$(X):NEXT

280 FORX=0TO21:READSX$(X):NEXT

290 B=0:C=0:CR=0:D=0:EX=0:J=0:M=0:E=0:N=
0:P=0:V=0:CH=0:L=0:BR=0

300 PRINT"[CLEAR]":COLOR0,1:COLOR4,1:COL
OR1,2

310 PRINT"[3 CR]*LONDON UNDERGROUND TUBE
 STATIONS*"

320 PRINT

330 FORX=0TO12:PRINTSPC(10):PRINTSX$(X):
NEXT

340 PRINTRR$;:PRINTSPC(12);"[2 CD]";

350 PRINTQQ$;"[3 SPACES]A-B"

360 PRINTSPC(12);QQ$;"[3 SPACES]B-C"

370 PRINTSPC(12);QQ$;"[3 SPACES]C-D-E"
```

```
1 REM ******************************
2 REM *                            *
3 REM * TUBE STATION INFORMATION *
4 REM *                            *
5 REM *       PLUS 4 ONLY          *
6 REM *                            *
7 REM * BY KEVIN WHEALS (C) 1989 *
8 REM ******************************

410 GOSUB4800:GRAPHIC0,1

420 DIMS$(270),SX$(21),A$(21),B$(21),C$(2
1),D$(21)

430 DIME$(21),F$(21),G$(21),H$(21)

440 DIMI$(21),J$(21),K$(21)

450 DIML$(21),Z$(13)

460 QQ$="[SPACE]STATIONS"

470 RR$="[HOME][1 CD]"

480 XX$="[1 CD][3 CR]PRESS A KEY TO RETU
N TO MAIN MENU"

490 UU$="[CLEAR][10 CR]UNDERGROUND STATIO
NS"

500 CL$="[CLEAR]**********UNDERGROUND ST
ATION**********[3 CD][10 CR]"

510 LL$="[3 CR]CHOOSE A KEY BETWEEN (0-9
) OR (A-L)"

520 BB$="[1 CD][5 CR][BROWN][REV,ON]BAKE
RLOO LINE[REV,OFF][WHITE]"

530 CC$="[1 CD][5 CR][RED][REV,ON]CENTRA
L LINE[REV,OFF][WHITE]"

540 CR$="[1 CD][5 CR][YELLOW][REV,ON]CIR
CLE LINE[REV,OFF][WHITE]"

550 DD$="[1 CD][5 CR][GREEN][REV,ON]DIST
RICT LINE[REV,OFF][WHITE]"

560 EX$="[1 CD][1 CR][GREEN][REV,ON]D[WH
ITE]I[GREEN]S[WHITE]T[GREEN]R[WHITE]I[GR
EEN]C[WHITE]T[GREEN][SPACE][WHITE]L[GREE
N]I[WHITE]N[GREEN]E[REV,OFF][WHITE]EXHIB
ITION SERVICE ONLY"
```

CON'T OVER!

```
530 IFZ$="3"THENGOTO1450

540 IFZ$="4"THENGOTO1720

550 IFZ$="5"THENGOTO1990

560 IFZ$="6"THENGOTO2260

570 IFZ$="7"THENGOTO2530

580 IFZ$="8"THENGOTO2800

590 IFZ$="9"THENGOTO3070

600 IFZ$="A"THENGOTO3340

610 IFZ$="B"THENGOTO3610

620 IFZ$="C"THENGOTO3880

630 GOTO290

640 GOSUB4020

650 FORY=0TO21:PRINTSPC(2);S$(Y):NEXT

660 PRINTLL$

670 GETA$:IFA$=""THEN670

680 IFA$="0"THENA$=S$(0):PRINTCL$;A$:GOTO
 4490

690 IFA$="1"THENA$=S$(1):PRINTCL$;A$:GOTO
 4410

700 IFA$="2"THENA$=S$(2):PRINTCL$;A$:GOTO
 4480
```

```
380 PRINTSPC(12);QQ$;"[3 SPACES]E-F-G"

390 PRINTSPC(12);QQ$;"[3 SPACES]G-H"

400 PRINTSPC(12);QQ$;"[3 SPACES]H-I-K-L"

410 PRINTSPC(12);QQ$;"[3 SPACES]L-M-N"

420 PRINTSPC(12);QQ$;"[3 SPACES]N-O-P"

430 PRINTSPC(12);QQ$;"[3 SPACES]P-Q-R-S"

440 PRINTSPC(12);QQ$;"[3 SPACES]S"

450 PRINTSPC(12);QQ$;"[3 SPACES]S-T-U-V"

460 PRINTSPC(12);QQ$;"[3 SPACES]V-W"

470 PRINTSPC(12);QQ$;"[3 SPACES]W"

480 PRINT:PRINTTAB(3);"CHOOSE A KEY BETW
EEN (0-9) OR (A-C)"

490 GETZ$:IFZ$=""THEN490

500 IFZ$="0"THENGOTO640

510 IFZ$="1"THENGOTO910

520 IFZ$="2"THENGOTO1180
```

CON'T
NEXT
MONTH!
ALONG WITH INFO!

Reader Letter

Dear Sir
Could you please let me know how to; PLOT: PRINT @ on the C16/+4.
These commands are used by my ORIC ATMOS and I would like to run them on the C16.

Yours faithfully
Peter Appleby.

Thanks for the letter Peter, well, here goes: PLOT on the C16/+4 would look like DRAW 1, X, Y, X & Y are screen coords, so DRAW 1, 10, 10 would PLOT a dot at 10, 10 (10 pixles down & 10 pixles across). As for PRINT @, well you could use CHAR , X, Y, "HELLO PETER", where X=ROW, Y=COLUMN, so you could have CHAR , 10, 10, "HELLO PETER", *which would display 'HELLO PETER'* 10 ROWS down

```
**********************************************************************
************************* SCREEN SCROLLING ***************************
**********************************************************************
First version.                BY PETER CRACK
*         This version is as printed in the ANCO book.
1010-1012 Set screen colour.#$C*=brightness,#$*F=colour.
1015-1034 Clear low-res.screen, IE.print space character (#$20) from
*         $0C00-$0FFF,clear colour area from $0800-$0BFF (#$00=fore ground
*         black).
1036-105F Print six lines of text,18 characters per line.
103B-103E Top line of top group and top line of bottom group.
1044-1047 Bottom line of top group and bottom line of bottom group.
104D      Top line of middle (scrolling) group.
1053      Bottom line of middle (scrolling) group.
1056-105B Colour for middle (scrolling) group,as before #$2*=brightness
*         #$*6=colour. Values for brightness range from #$0* to #$7* and
*         #$8* to #$F* for the same brightness but flashing,in the same byte
*         colour ranges from #$*0 to #$*F for colour,to explain and
*         demonstrate:- try putting #$a9 into $1057..>1056 A9 A9 LDA #$A9.
1061-1063 Clear $D0 and $DB remember 'X' register was set to zero in $1015
*         and not changed since.
1065      Set new interupt routine (used only once ).
1068      Start of main routine.
106B      #$FD=JOY(1),#$FA=JOY(2) or maybey the other way around,I never can
*         remember!.
106D-108D Get joystick return.
106D-1070 Keyboard latch!!!.$FF09 contains joystick values in inverse logic,
*         that is to say,if joystick is moved then the bit which registers
*         the movement is set to zero IE:-
*         BIT NUMBER 7  6  5  4  3  2  1  0
*         BIT VALUE  1  1  1  1  1  1  1  1
*         All bits set to 1 or high=joystick not moved
*         Bit 0=0 joystick moved up
*         Bit 1=0 joystick moved down
*         Bit 2=0 joystick moved left
*         Bit 3=0 joystick moved right
*         Bit 7=0 joystick fire button pressed
*         Bits 4,5 and 6 not used for joystick returns,COMMODORES joystick
*         return <BASIC RJOY(x)> can be viewed at $BFC1-$BFFC in ROM,routine
*         at $BFFD-$BFEC flips (invert with'EOR'command) all bits and then
*         evaluates to give 'CORRECT' returns of 0-8 for direction and adds
*         DEC 128 if fire button was pressed.Unfortunately this,COMMODORES,
*         routine is not very accurate when moving diagonaly or moving and
*         firing at the same time. (or else my joystick is at fault!).
1073      Clear 'X' register.
1075-1076 Discard up and down bits.
1077-107A Check move left and decrease 'X' register if so.
107B-107E Do the same for move right.
107F-1081 Transfer 'X' register to accumulator and add present movement
*         speed,register.
1083-1089 Check against speed limits at $1083 and $1087 discard if too fast
*         either left or right (negative or positive speeds).
108B      Update speed register if values within limmits.
108D      And force branch to $1068.
108F-1097 Disable interupts,reset $0314/$0315 to point to your new routine.
109A-109C Set interupt register to request interupts from raster possition.
109F-10A3 Clear $DA (end of line pointer) and $D9 (scroll counter).
10A5-10B7 Set first raster interupt possition,this is the start of the
*         screen scroll area.screen raster possitions run from #$00 to #$C8
*         #$C9-#$FF cannot be seen,(or used I think).
10AA-10AC Set horizontal scroll possition to zero,and,by clearing bit 3,of
*         $FF07 reduce screen from 40 columns to 38.
10AF-10B0 Clear interupt disable (opposite of $108F) and return
10B1-1173 Main routine for moving centre (scrolling) text
10B1-10B3 Wait here until your interupt routine has placed a non-zero value
*         into $DB.....CONTINUED.....
```

10B5-10B7 Clear $DB,this is to ensure that no text is scrolled until screen
* raster beam has moved away from scrolling area.
10B9-10BC Load last scroll position ($D9),add speed register ($D0),this sum
* becomes new amount of scrolling required to maintain,increase or
* decrease scrolling speed.
10BE Transfer accumulator to 'Y' register for safekeeping
10BF-10C1 Discard all but the rightmost three bits in accumulator and store
* in $D9,this is now the new scroll position for the screen.
10C3 Transfer the 'Y' register back to the accumulator
10C4-10C8 Check to see if bit three is set (does the text need to be moved
* one character square left or right) if 'yes' then branch to $10C9
* else return from subroutine.
10C9 Load 'X' register with the number of screen lines to be scrolled.
10CB-10D9 Set $D2,$D4 to start address (high byte) of scrolling area(char
* ram),set $D6,$D8 to same for colour area.Set $D1 to point to start
* address (low byte) of the left hand end of the line directly above
* scrolling area.
10DB 10DB Transfer 'Y' register to accumulator,remember 'Y' register still
* contains the result of the addition of $D9 and $D0 $10B9-$10BC.
10DC 10DC Is it positive or negative EG. 0-126 or 128-255 EG. #$00-#$7F or
* #$FF-#$80 (in this case we want to know do we have to move right
10DE one character or left one character.
10DE-110D Move left routine
10DE 10DE Load 'Y' register with #$FF=dec-1,this is so that at $10FB when
10E0 the 'Y' register is increased by one it starts at zero (#$FF).
10E0-10EB Load preset $D1 register,add #$28 (dec 40) to move it down one
line to the start position of scrolling area,store it in $D1,$D3,
$D5 and $D7,these are the low bytes of the start points for
10E0 character and colour of the scrolling areas.
10E0-10F7 Check to see if this addition has rolled $D1-$D7 over #$FF(dec255)
thus setting the carry flag,if 'yes' then increase $D2,$D4,$D6 and
10F7 $D8,these are the rlevant high bytes of the start addresses.
10F7-10F9 Increase $D3 and $D7 low bytes of character and colour addresses
10FB to point to the character one to the right of start addres
10FB Increase 'Y' register ($10FB-$1104) move one line routine.
10FC 10FC Load character one to right of start point (for this line) offset
by 'Y' register.
10FE 10FE Store this character in start point (for this line) offset by 'Y'
register.
10FE 10FE Load colour of character one to right of start point (for this
line), offset by 'Y' register.
1102 1102 Store it in start point (offset by 'Y' register).
1104 1104 Is the whole line done?,no,well branch to $10FB else.....
1108 1108 Are all the lines done?,no,then branch to $10DE else.....
1108-110D Store 'Y' register in $DA ('Y' register now=#$27=end of line) and
110B force branch to $113C.
110F
110F-113A As above but move right one character.
110F 110F Load 'Y' register with offset #$26 (points to one character to the
1111 left of right hand side of the screen scrolling area.
1111-112A Load $D1,add #$28 and set $D1-$D7 as before.
112C-1138 Reverse of routine at $10FC-$1102.
112C 112C Load character at the right hand end of the line (all loading and
storing is offset by the 'Y' register)
112E 112E Store it in the character position which is hidden. Remember,that
although you can only see 38 characters per screen line 40 can
113C still be used.
1130-1132 Do the same for colour data.
113C
1134-1135 Do it for the whole line.
1137
1137-1138 Do it for all the scrolling lines.
113A 113A Clear $DA,the previous command ensures that 'X' register at this
113C point will always=#$00.
113C-1172 This routine takes the end (hidden) character from one side of the
screen and puts it at the other end (hidden),the value in $DA is
either #$00 or #$27 depending wether you are scrolling left or
right respectively.....CONTINUED.....

```
113C        Load 'X' register with the number of lines to scroll
113E        Load 'Y' register with $DA,that is #$00 or #$27 depending on
*           direction of scroll.
1140-114C   Set $D2,$D6,$D1 and $D5 to point to start of line directly above
*           first line of scrolling area for characters and colour.
114E-115B   Add #$28 (in effect move one line down lo-res screen) to $D1 and
*           $D5,check if they have rolled over #$FF or dec 255 if yes then
*           increase $D2 and $D6.
115D-1160   This is the tricky bit,because we have now,to set 'Y' register to
*           point to the character which has just moved into the hidden column
*           of the screen ($DA is set to point to the other end) we know only
*           two value are correct #$00 is the left end and #$27 is the right,
*           we could use two separate routine but,to save space this method
*           is better,first transfer a copy of 'Y' register to the accumulator
*           create a mask number,(command EOR #$27) flip the relevant bits in
*           the accumulator and store the new result back into the 'Y'register
*           see fig1 for explanation of EOR command.
1161        Load a character from this line offset by 'Y' register.
1163        Store it in a temporary register ($DC).
1165        Load colour data for same character offset by 'Y' register.
1167        Load 'Y' register with $DA (opposite end of line).
1169        Store colour data back into screen offset by new 'Y'register value
116B-116D   Do the same for character,at this point the same character will
*           appear at both ends of the screen line,IE.column 0 and column 40
*           but,as both these columns are hidden it does not matter.
116F-1170   Do the same for all the lines.
1173-119E   this is the new interupt routine,this routine does not service
*           $CE0E,in other words the keyboard and all other input/output
*           routines including checking and decreasing sound counters (note
*           duration) is ignored therefore the only way to stop this programme
*           is to press the reset button or run/stop and reset buttons
*           together.
1173-1176   Read and clear interupt register(register cleared by STA command).
1179        Load 'X' register with last position whre you wanted to request an
*           interupt.
117C        Load accumulator with #$40,why this value was chosen I do not know
*           the important thing is that the four rightmost bits are set to
*           zero so the four leftmost could be any value,as later on they
*           will be discarded.
117E        was this interupt requested at the bottom of scroll area?.
1180        If yes then branch to $118A.
1182        if no then mix the value of $D9 into the accumulator.
1184-1188   Load 'X' register with bottom of scroll area,store it in $DB,(this
*           will ensure that the routine at $11B1 will be carried out) and
*           branch to $118C.
118A        Load 'X' register with top of scroll area.
118C        and store it in raster comparison register.(this register will
*           request an interupt every time the raster beam reaches the
*           position set in it).
118F        Discard all but the rightmost three bits of the accumulator,
*           remember the accumulator holds the sum of value placed into it at
*           $117C and the value mixed into it at $1182 and has not been
*           changed until now,the screen can only be offset 8 pixel points to
*           the right and the values 0-7 are the maximum combination of values
*           obtained from 3 bits and it is the 3 rightmost bits of $FF07 that
*           controll this screen offset.
1191-1194   Halt programme execution here if the raster beam position value is
*           the same as the raster comparison value.(We do not want another
*           interupt until this one is finnished!!! because the computer will
*           probably lock up).
1196        Set the screen offset.$FF07 also controls other actions IE:-
*           bit 3=0=38 columns,bit 3=1=40 columns,bit 4=0=MCM mode off,
*           bit 4=1=MCM mode on,bits 5-7 controll freeze mode(which means just
*           that,both screen and computer switch off),PAL/NTSC mode two types
*           of TV.signal,.....CONTINUED.....
```

```
**********************************************************************
*         and reverse video mode respctively,these bits must all be set high
*         (set to 1) to be active,in this case we do not want any of them.
1199-119E These commands pull the old 'Y','X' and accumulator values off the
*         stack,the last command,RTI tells the computer that an interupt has
*         just been serviced and also acts as an RTS command,(return from
*         subroutine) to return controll to your main programme.This section
*         reverses the routine at $FCB3-$FCB7 and replaces $FCB8-$FCC8.
119F-11EE Data for lines of text,add #$40 to each value to read message
Having read through all this do not,unless you are a genious,expect to
Understand it in one go (I have had this book two years) just compare these
Notes with the listing and go over it again and again and again etc....zzzzz
FIG 1.
Accumulator=#$FF.Decimal 255
BIT number.............7   6   5   4   3   2   1   0
BIT state..............1   1   1   1   1   1   1   1
BIT hex value.........80  40  20  10 08  04  02  01
BIT decimal value.....128 64  32  16  8   4   2   1
Explanation of command at $115E..
Accumulator=#$00 (zero) at start  BIT state 0 0 0 0 0 0 0 0
EOR (exclusive-OR) mask=#$27..... BIT state 0 0 1 0 0 1 1 1
Accumulator after comparison..... BIT state 0 0 1 0 0 1 1 1
EOR command flips (changes the state of) all the bits in the accumulator
Whose twins had been set to one in the EOR mask.This is a rollover type
Command and if used twice will return the accumulator to its previous value.
**********************************************************************
THE FOLLOWING EXPLANATION REFERS TO MY VERSION OF THE SAME PROGRAMME LINES
**********************************************************************
$1200-$1350,only the main changes have been noted.
1200-120C Set screen colour,set forground colour,clear lo-res screen.
120F-121B Set $D0-$D3 to point to about middle of top line of screen.
121D      Set 'X' register with number of characters to print.
121F-1239 Print 25 'A' characters down the screen,colour dark green.
125D-1261 Speed limits reduced to prevent flicker on screen.
127F-1281 Set start point of screen scroll area in this case top of screen.
1300-1306 Lower limit of scroll area (bottom of visible screen).
130C      Upper limit of scrolling area.
1321-1333 Set number of lines to scroll in 'X' register,set $D1-$D8 to left
*         hand end of the line above the start point of the scrolling area.
1334-134F Moves all the relevant pointers down one line.
**********************************************************************
**********************************************************************
BOTH THE ANCO AND MY PROGRAMME ARE ESSENTIALY THE SAME MINE IS A LITTLE MORE
OSCURE DUE TO THE USE OF SUBROUTINES WHICH REDUCES THE NUMBER OF LINES.
**********************************************************************
How to use in your own programme???.
**********************************************************************
That is of course up to you!.
But here are two suggestions,first make the whole programme interupt
Controlled entering at $1068 or,second and probably more practical,make
It a subroutine entering at $1068 and leaving at $108D.In either case I
Would suggest putting JMP$CE0E at $1199 so that a normal interupt can be
Serviced,unless you would prefer to write your own keyboard etc.check.
*
*
As always any problems ring me or better still write in to the magazine.....
*
*
*.....................Good luck.............................PETER.............
**********************************************************************
```

```
. 1010  A9 CF     LDA #$CF
. 1012  8D 15 FF  STA $FF15
. 1015  A2 00     LDX #$00
. 1017  A9 20     LDA #$20
. 1019  9D 00 0C  STA $0C00,X
. 101C  9D 00 0D  STA $0D00,X
. 101F  9D 00 0E  STA $0E00,X
. 1022  9D 00 0F  STA $0F00,X
. 1025  A9 00     LDA #$00
. 1027  9D 00 08  STA $0800,X
. 102A  9D 00 09  STA $0900,X
. 102D  9D 00 0A  STA $0A00,X
. 1030  9D 00 0B  STA $0B00,X
. 1033  CA        DEX
. 1034  D0 E1     BNE $1017
. 1036  A0 12     LDY #$12
. 1038  B9 9F 11  LDA $119F,Y
. 103B  99 82 0C  STA $0C82,Y
. 103E  99 2A 0F  STA $0F2A,Y
. 1041  B9 B2 11  LDA $11B2,Y
. 1044  99 D2 0C  STA $0CD2,Y
. 1047  99 7A 0F  STA $0F7A,Y
. 104A  B9 C5 11  LDA $11C5,Y
. 104D  99 C2 0D  STA $0DC2,Y
. 1050  B9 D8 11  LDA $11D8,Y
. 1053  99 62 0E  STA $0E62,Y
. 1056  A9 26     LDA #$26
. 1058  99 C2 09  STA $09C2,Y
. 105B  99 62 0A  STA $0A62,Y
. 105E  88        DEY
. 105F  10 D7     BPL $1038
. 1061  86 D0     STX $D0
. 1063  86 DB     STX $DB
. 1065  20 8F 10  JSR $108F
. 1068  20 B1 10  JSR $10B1
. 106B  A9 FD     LDA #$FD
. 106D  8D 08 FF  STA $FF08
. 1070  AD 08 FF  LDA $FF08
. 1073  A2 00     LDX #$00
. 1075  4A        LSR
. 1076  4A        LSR
. 1077  4A        LSR
. 1078  B0 01     BCS $107B
. 107A  CA        DEX
. 107B  4A        LSR
. 107C  B0 01     BCS $107F
. 107E  E8        INX
. 107F  8A        TXA
. 1080  18        CLC
. 1081  65 D0     ADC $D0
. 1083  C9 F7     CMP #$F7
. 1085  F0 E1     BEQ $1068
. 1087  C9 09     CMP #$09
. 1089  F0 DD     BEQ $1068
. 108B  85 D0     STA $D0
. 108D  D0 D9     BNE $1068
. 108F  78        SEI
. 1090  A9 73     LDA #$73
. 1092  8D 14 03  STA $0314
. 1095  A9 11     LDA #$11
. 1097  8D 15 03  STA $0315
. 109A  A9 02     LDA #$02
. 109C  8D 0A FF  STA $FF0A
. 109F  A9 00     LDA #$00
. 10A1  85 DA     STA $DA
. 10A3  85 D9     STA $D9
. 10A5  A9 5A     LDA #$5A
. 10A7  8D 0B FF  STA $FF0B
. 10AA  A9 00     LDA #$00
. 10AC  8D 07 FF  STA $FF07
. 10AF  58        CLI
. 10B0  60        RTS
. 10B1  A5 DB     LDA $DB
. 10B3  F0 FC     BEQ $10B1
. 10B5  A9 00     LDA #$00
. 10B7  85 DB     STA $DB
. 10B9  A5 D9     LDA $D9
. 10BB  18        CLC
. 10BC  65 D0     ADC $D0
. 10BE  A8        TAY
. 10BF  29 07     AND #$07
. 10C1  85 D9     STA $D9
. 10C3  98        TYA
. 10C4  29 08     AND #$08
. 10C6  D0 01     BNE $10C9
. 10C8  60        RTS
. 10C9  A2 05     LDX #$05
. 10CB  A9 0D     LDA #$0D
. 10CD  85 D2     STA $D2
. 10CF  85 D4     STA $D4
. 10D1  A9 09     LDA #$09
. 10D3  85 D6     STA $D6
. 10D5  85 D8     STA $D8
. 10D7  A9 90     LDA #$90
. 10D9  85 D1     STA $D1
. 10DB  98        TYA
. 10DC  10 31     BPL $110F
. 10DE  A0 FF     LDY #$FF
. 10E0  A5 D1     LDA $D1
. 10E2  18        CLC
. 10E3  69 28     ADC #$28
. 10E5  85 D1     STA $D1
. 10E7  85 D3     STA $D3
. 10E9  85 D5     STA $D5
. 10EB  85 D7     STA $D7
. 10ED  90 08     BCC $10F7
. 10EF  E6 D2     INC $D2
. 10F1  E6 D4     INC $D4
. 10F3  E6 D6     INC $D6
. 10F5  E6 D8     INC $D8
. 10F7  E6 D3     INC $D3
. 10F9  E6 D7     INC $D7
. 10FB  C8        INY
. 10FC  B1 D3     LDA ($D3),Y
. 10FE  91 D1     STA ($D1),Y
. 1100  B1 D7     LDA ($D7),Y
. 1102  91 D5     STA ($D5),Y
. 1104  C0 27     CPY #$27
. 1106  D0 F3     BNE $10FB
. 1108  CA        DEX
. 1109  D0 D3     BNE $10DE
. 110B  84 DA     STY $DA
. 110D  F0 2D     BEQ $113C
. 110F  A0 26     LDY #$26
. 1111  A5 D1     LDA $D1
. 1113  18        CLC
. 1114  69 28     ADC #$28
. 1116  85 D1     STA $D1
. 1118  85 D3     STA $D3
. 111A  85 D5     STA $D5
. 111C  85 D7     STA $D7
. 111E  90 08     BCC $1128
```

112A E6 D2
INC $D2

```
1122  E6 D4     INC $D4
1124  E6 D6     INC $D6
1126  E6 D8     INC $D8
1128  E6 D3     INC $D3
112A  E6 D7     INC $D7
112C  B1 D1     LDA ($D1),Y
112E  91 D3     STA ($D3),Y
1130  B1 D5     LDA ($D5),Y
1132  91 D7     STA ($D7),Y
1134  88        DEY
1135  10 F5     BPL $112C
1137  CA        DEX
1138  D0 D5     BNE $110F
113A  86 DA     STX $DA
113C  A2 05     LDX #$05
113E  A4 DA     LDY $DA
1140  A9 0D     LDA #$0D
1142  85 D2     STA $D2
1144  A9 09     LDA #$09
1146  85 D6     STA $D6
1148  A9 90     LDA #$90
114A  85 D1     STA $D1
114C  85 D5     STA $D5
114E  A5 D1     LDA $D1
1150  18        CLC
1151  69 28     ADC #$28
1153  85 D1     STA $D1
1155  85 D5     STA $D5
1157  90 04     BCC $115D
1159  E6 D2     INC $D2
115B  E6 D6     INC $D6
115D  98        TYA
115E  49 27     EOR #$27
1160  A8        TAY
1161  B1 D1     LDA ($D1),Y
1163  85 DC     STA $DC
1165  B1 D5     LDA ($D5),Y
1167  A4 DA     LDY $DA
1169  91 D5     STA ($D5),Y
116B  A5 DC     LDA $DC
116D  91 D1     STA ($D1),Y
116F  CA        DEX
1170  D0 DC     BNE $114E
1172  60        RTS
1173  AD 09 FF  LDA $FF09
1176  8D 09 FF  STA $FF09
1179  AE 0B FF  LDX $FF0B
117C  A9 40     LDA #$40
117E  E0 83     CPX #$83
1180  F0 08     BEQ $118A
1182  05 D9     ORA $D9
1184  A2 83     LDX #$83
1186  86 DB     STX $DB
1188  D0 02     BNE $118C
118A  A2 5A     LDX #$5A
118C  8E 0B FF  STX $FF0B
118F  29 07     AND #$07
1191  EC 1D FF  CPX $FF1D
1194  F0 FB     BEQ $1191
1196  8D 07 FF  STA $FF07
1199  68        PLA
119A  A8        TAY
119B  68        PLA
119C  AA        TAX
119D  68        PLA
119E  40        RTI
```

```
DATA FOR ANCO AND MY LISTINGS.....
>119F 20 04 09 05 13 05 12 20 : .....
>11A7 14 05 09 0C 20 08 09 05 :.... ...
>11AF 12 20 20 20 20 20 02 0C :.     ..
>11B7 05 09 02 14 20 13 14 05 :.... ...
>11BF 08 05 0E 20 20 20 16 05 :...   ..
>11C7 12 13 03 08 09 05 02 05 :........
>11CF 20 04 01 13 20 08 09 05 : ... ...
>11D7 12 0D 09 14 20 05 09 0E :.... ...
>11DF 05 0D 20 0A 0F 19 13 14 :.. .....
>11E7 09 03 0B 21 AA AA AA AA :...!****
>11EF EA EA EA EA EA EA EA EA :JJJJJJJJ
>11F7 EA EA EA EA EA EA EA EA :JJJJJJJJ
```

```
.....START OF MY LISTING.....
. 1200  A9 CF     LDA #$CF
. 1202  8D 15 FF  STA $FF15
. 1205  A9 00     LDA #$00
. 1207  8D 3B 05  STA $053B
. 120A  A9 93     LDA #$93
. 120C  20 D2 FF  JSR $FFD2
. 120F  A9 0C     LDA #$0C
. 1211  85 D1     STA $D1
. 1213  A9 10     LDA #$10
. 1215  85 D0     STA $D0
. 1217  85 D2     STA $D2
. 1219  A9 08     LDA #$08
. 121B  85 D3     STA $D3
. 121D  A2 19     LDX #$19
. 121F  A0 00     LDY #$00
. 1221  A9 01     LDA #$01
. 1223  91 D0     STA ($D0),Y
. 1225  A9 0F     LDA #$0F
. 1227  91 D2     STA ($D2),Y
. 1229  A5 D0     LDA $D0
. 122B  18        CLC
. 122C  69 29     ADC #$29
. 122E  90 04     BCC $1234
. 1230  E6 D1     INC $D1
. 1232  E6 D3     INC $D3
. 1234  85 D0     STA $D0
. 1236  85 D2     STA $D2
. 1238  CA        DEX
. 1239  D0 E6     BNE $1221
. 123B  86 D0     STX $D0
. 123D  86 DB     STX $DB
. 123F  20 69 12  JSR $1269
. 1242  20 8B 12  JSR $128B
. 1245  A9 FD     LDA #$FD
. 1247  8D 08 FF  STA $FF08
. 124A  AD 08 FF  LDA $FF08
. 124D  A2 00     LDX #$00
. 124F  4A        LSR
. 1250  4A        LSR
. 1251  4A        LSR
. 1252  B0 01     BCS $1255
. 1254  CA        DEX
. 1255  4A        LSR
. 1256  B0 01     BCS $1259
. 1258  E8        INX
. 1259  8A        TXA
. 125A  18        CLC
```

CONTINUED.....

```
. 125B  65 D0     ADC $D0
. 125D  C9 FB     CMP #$FB
. 125F  F0 E1     BEQ $1242
. 1261  C9 04     CMP #$04
. 1263  F0 DD     BEQ $1242
. 1265  85 D0     STA $D0
. 1267  D0 D9     BNE $1242
. 1269  78        SEI
. 126A  A9 F5     LDA #$F5
. 126C  8D 14 03  STA $0314
. 126F  A9 12     LDA #$12
. 1271  8D 15 03  STA $0315
. 1274  A9 02     LDA #$02
. 1276  8D 0A FF  STA $FF0A
. 1279  A9 00     LDA #$00
. 127B  85 DA     STA $DA
. 127D  85 D9     STA $D9
. 127F  A9 00     LDA #$00
. 1281  8D 0B FF  STA $FF0B
. 1284  A9 00     LDA #$00
. 1286  8D 07 FF  STA $FF07
. 1289  58        CLI
. 128A  60        RTS
. 128B  A5 DB     LDA $DB
. 128D  F0 FC     BEQ $128B
. 128F  A9 00     LDA #$00
. 1291  85 DB     STA $DB
. 1293  A5 D9     LDA $D9
. 1295  18        CLC
. 1296  65 D0     ADC $D0
. 1298  A8        TAY
. 1299  29 07     AND #$07
. 129B  85 D9     STA $D9
. 129D  98        TYA
. 129E  29 08     AND #$08
. 12A0  D0 01     BNE $12A3
. 12A2  60        RTS
. 12A3  20 21 13  JSR $1321
. 12A6  98        TYA
. 12A7  10 19     BPL $12C2
. 12A9  A0 FF     LDY #$FF
. 12AB  20 34 13  JSR $1334
. 12AE  C8        INY
. 12AF  B1 D3     LDA ($D3),Y
. 12B1  91 D1     STA ($D1),Y
. 12B3  B1 D7     LDA ($D7),Y
. 12B5  91 D5     STA ($D5),Y
. 12B7  C0 27     CPY #$27
. 12B9  D0 F3     BNE $12AE
. 12BB  CA        DEX
. 12BC  D0 EB     BNE $12A9
. 12BE  84 DA     STY $DA
. 12C0  F0 15     BEQ $12D7
. 12C2  A0 26     LDY #$26
. 12C4  20 34 13  JSR $1334
. 12C7  B1 D1     LDA ($D1),Y
. 12C9  91 D3     STA ($D3),Y
. 12CB  B1 D5     LDA ($D5),Y
. 12CD  91 D7     STA ($D7),Y
. 12CF  88        DEY
. 12D0  10 F5     BPL $12C7
. 12D2  CA        DEX
. 12D3  D0 ED     BNE $12C2
. 12D5  86 DA     STX $DA
. 12D7  20 21 13  JSR $1321
. 12DA  A4 DA     LDY $DA
. 12DC  20 34 13  JSR $1334
. 12DF  98        TYA
. 12E0  49 27     EOR #$27
. 12E2  A8        TAY
. 12E3  B1 D1     LDA ($D1),Y
. 12E5  85 DC     STA $DC
. 12E7  B1 D5     LDA ($D5),Y
. 12E9  A4 DA     LDY $DA
. 12EB  91 D5     STA ($D5),Y
. 12ED  A5 DC     LDA $DC
. 12EF  91 D1     STA ($D1),Y
. 12F1  CA        DEX
. 12F2  D0 E8     BNE $12DC
. 12F4  60        RTS
. 12F5  AD 09 FF  LDA $FF09
. 12F8  8D 09 FF  STA $FF09
. 12FB  AE 0B FF  LDX $FF0B
. 12FE  A9 00     LDA #$00
. 1300  E0 CA     CPX #$CA
. 1302  F0 08     BEQ $130C
. 1304  05 D9     ORA $D9
. 1306  A2 CA     LDX #$CA
. 1308  86 DB     STX $DB
. 130A  D0 02     BNE $130E
. 130C  A2 00     LDX #$00
. 130E  8E 0B FF  STX $FF0B
. 1311  29 07     AND #$07
. 1313  EC 1D FF  CPX $FF1D
. 1316  F0 FB     BEQ $1313
. 1318  8D 07 FF  STA $FF07
. 131B  68        PLA
. 131C  A8        TAY
. 131D  68        PLA
. 131E  AA        TAX
. 131F  68        PLA
. 1320  40        RTI
. 1321  A2 19     LDX #$19
. 1323  A9 0B     LDA #$0B
. 1325  85 D2     STA $D2
. 1327  85 D4     STA $D4
. 1329  A9 07     LDA #$07
. 132B  85 D6     STA $D6
. 132D  85 D8     STA $D8
. 132F  A9 D8     LDA #$D8
. 1331  85 D1     STA $D1
. 1333  60        RTS
. 1334  A5 D1     LDA $D1
. 1336  18        CLC
. 1337  69 28     ADC #$28
. 1339  85 D1     STA $D1
. 133B  85 D3     STA $D3
. 133D  85 D5     STA $D5
. 133F  85 D7     STA $D7
. 1341  90 0B     BCC $134B
. 1343  E6 D2     INC $D2
. 1345  E6 D4     INC $D4
. 1347  E6 D6     INC $D6
. 1349  E6 D8     INC $D8
. 134B  E6 D3     INC $D3
. 134D  E6 D7     INC $D7
. 134F  60        RTS
. 1350  EA        NOP
```

END OF MY LISTING

FOR SALE & WANTED PAGE

WANTED: —

Would like to buy, DISK DRIVE, Model 1551, Any-Offers ?

Ring 0493-730963, and ask for Kevin or write to Kevin at:-
Kevin Williams, 10 Hickling Way, Ormesby St Margaret, Gt Yarmouth, NORFOLK, NR29 3SE. (Kevin, do you still want this ad in, call me please, THANKS!!)

WANTED: —

Any old Broken/Working C16/+4 harware, ie, Joytsicks, tapedecks, D/D etc, must be cheap, please contact:-
Roy Robinson, 112 Cliff Road, HORNSEA, N. Humberside, HU18 1JE.
Tel (0964) 534611

FOR SALE: —

C16/Plus/4 Printer Service.

Have you got Programs, Letters etc you want printing, but cannot afford a printer?

Well worry no more because C16/Plus/4 Printer Service is here! We can print out Basic Files from tape or disk, Basic programs saved with Turbo-Plus. Wordprocessor programs from 3+1 or Script-Plus. We can also print high/low res Graphic dumps from your programs (NOT PROTECTED COMMERCIAL ONES).

This service is exclusive to members of this Club and it costs only 60p for the first copy and 15p for any other copies there after. This price INCLUDES return postage and packing.

Please send your tape/disk in a jiffy bag/disk mailer along with money and amount required to:-
Plus/4 Printer Service, Daniel Stokes, 35 Burleigh Way, CUFFLEY, Herts, EN6 4LG.

FOR SALE: —

C64, Datarecorder, PSU, Loads of games too many to mention here. Contact:
Mr W.D. Brighton, 55B Occupation Lane, SHEFFIELD, S12 4PS.

Telephone 0742-641046

WANTED: —

3764 RAM CHIPS OR 4164 RAM CHIPS OR 4564 RAM CHIPS, I need 8 of any of the listed chips, I will pay £5 for them, desoldered etc, contact: John Hadlow, Showground, Buchan Park, Greendykes Road, Broxburn, W. Lothian, SCOTLAND.

TERRA NOVA GAME TIP
FROM MATTEW NEWTON-LEWIS, W-SUSSEX.

While playing, press RUN/STOP, this will pause the game. After waiting a while press '1'(ONE) on the keyboard to start with infinite lives, NOT BAD EH ?